Cloud Robotics: Architecture, Challenges and Applications

Article *in* IEEE Network · May 2012

DOI: 10.1109/MNET.2012.6201212

3 authors:

Guoqiang Hu
Nanyang Technological University
198 PUBLICATIONS **3,986** CITATIONS

SEE PROFILE

Wee Peng Tay
Nanyang Technological University
156 PUBLICATIONS **1,953** CITATIONS

SEE PROFILE

Yonggang Wen
Nanyang Technological University
248 PUBLICATIONS **6,394** CITATIONS

SEE PROFILE

Cloud Robotics: Architecture, Challenges and Applications

Guoqiang Hu, Wee Peng Tay, and Yonggang Wen, Nanyang Technological University

Abstract

We extend the computation and information sharing capabilities of networked robotics by proposing a cloud robotic architecture. The cloud robotic architecture leverages the combination of an ad-hoc cloud formed by machine-to-machine (M2M) communications among participating robots, and an infrastructure cloud enabled by machine-to-cloud (M2C) communications. Cloud robotics utilizes an elastic computing model, in which resources are dynamically allocated from a shared resource pool in the ubiquitous cloud, to support task offloading and information sharing in robotic applications. We propose and evaluate communication protocols, and several elastic computing models to handle different applications. We discuss the technical challenges in computation, communications and security, and illustrate the potential benefits of cloud robotics in different applications.

Robotic systems have brought significant socioeconomic impacts to human lives over the past few decades [1]. For example, industrial robots (especially robot manipulators) have been widely deployed in factories to do tedious, repetitive, or dangerous tasks, such as assembly, painting, packaging, and welding. These preprogrammed robots have been very successful in industrial applications due to their high endurance, speed, and precision in structured factory environments. To extend the functional range of these robots or to deploy them in unstructured environments, robotic technologies are integrated with network technologies to foster the emergence of networked robotics.

A networked robotic system refers to a group of robotic devices that are connected via a wired and/or wireless communication network [2]. Networked robotics applications can be classified as either *teleoperated* robots or *multi-robot* systems. In the former case, a human operator controls or manipulates a robot at a distance by sending commands and receiving measurements via the communication network. Application examples include remote control of a planetary rover and remote medical surgery. In the latter case, a team of networked robots complete a task cooperatively in a distributed fashion by exchanging sensing data and information via the communication network. Examples include cooperative robot manipulators, a team of networked robots performing search and rescue missions, and a group of micro satellites working cooperatively in a desired formation.

Networked robotics, similar to standalone robots, faces inherent physical constraints as all computations are conducted onboard the robots, which have limited computing capabilities. Information access is also restricted to the collective storage of the network. With the rapid advancement of wireless communications and recent innovations in cloud computing technologies, some of these constraints can be overcome through the concept of *cloud robotics*, leading to more intelligent, efficient and yet cheaper robotic networks. In this article, we describe a cloud robotics architecture, some of the technical challenges, and its potential applications. Some preliminary results on the optimal operation of cloud robotics are also presented.

The rest of the article is organized as follows. First, we outline various challenges and constraints in networked robotics. Next, we describe the cloud robotics architecture, and elaborate on two key enabling sub-systems. Follwing that we address technical challenges in designing and operating the cloud robotics architecture. We will also highlight a few important robotic applications that will benefit from the cloud robotics. Finally, we conclude and summarize this article.

Challenges in Networked Robotics

Networked robotics, especially the multi-robot system as illustrated in Fig. 1, distributes the workload of sensing, actuating, communication, and computation among a group of participating robots. It has achieved great success in industrial applications, intelligent transportation systems, and security applications. However, the advancement of networked robotics is restricted by resource, information, and communication constraints inherent in the existing framework. We discuss these constraints in detail in this section.

Resource Constraints

Although a robot can share its computation workload with other units in the formation, the overall effectiveness of the robotic network is limited by the collection of each robot's resources, including onboard computers or embedded computing units, memories, and storage space. Physically, these onboard computing devices are restricted by the robots' size, shape, power supply, motion mode, and working environment. Once the robots are designed, built and deployed, it is technically challenging, if not infeasible, to change or upgrade their resource configurations.

0890-8044/12/$25.00 © 2012 IEEE

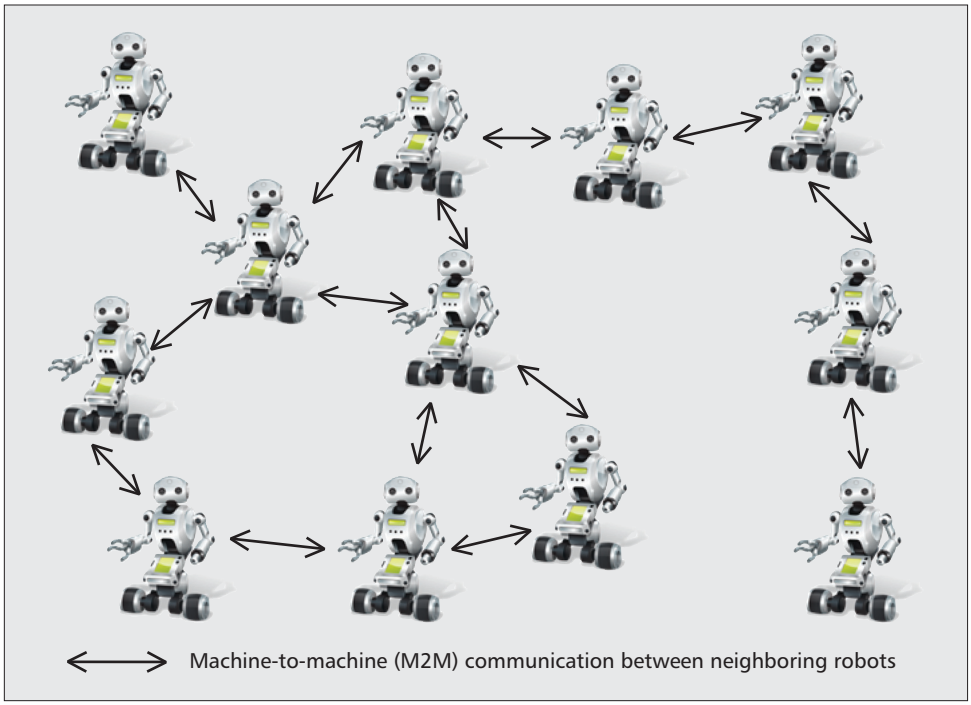

Figure 1. *Networked Robotics: a team of robots are interconnected with a communication network to collaboratively accomplish tasks.*

Information and Learning Constraints

The amount of information a robot has access to is constrained by its processing power, storage space, and the number and type of sensors it carries. Networked robotics allows the sharing of information amongst robots connected by a communication network so that a global task can be solved or computed cooperatively using the whole network. However, networked robotics is constrained by the information observed or computed by robots in the network, and by the examples or scenarios that the network encounters, and hence limiting its ability to learn. A robotic team learning to navigate may perform very well in a static environment, where all obstacles can be mapped out with increasing accuracy over time. On the other hand, the learning process has to be repeated once the environment changes or the robotic team is placed in a new unfamiliar environment. The map databases maintained by the robotic formation is also limited by the collective amount of storage space (including memory and disk) the formation has.

Communication Constraints

Common protocols for machine-to-machine (M2M) communications include *proactive routing*, which involves the periodic exchange of messages so that routes to every possible destination in the network are maintained [3], and *ad-hoc routing*, which forms a dynamic route to a destination node only when there is a message to be sent [4]. Proactive routing incurs high computation and memory resources in the route discovery and maintenance process. Ad-hoc routing protocols suffer from a high latency as a route has to be established before a message can be sent, and are not practical if the network topology is highly dynamic. These drawbacks are significant in mobile robotic networks, and may lead to severe performance degradation.

From Networked Robotics to Cloud Robotics

Networked robotics can be considered as an *evolutionary step towards cloud robotics*, i.e., *cloud-enabled networked robotics*, which leverages emerging cloud computing technologies to transform networked robotics. The design objective is to overcome the limitations of networked robotics with elastic resources offered by a ubiquitous cloud infrastructure.

Cloud computing provides a natural venue to extend the capabilities of networked robotics. NIST [5] defines *cloud computing* as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction." Through its three service models (i.e., software, platform and infrastructure), it enables tremendous flexibility in designing and implementing new applications for networked robotics.

Several research groups have started to explore the use of cloud technologies in robotic applications. For example, research groups at Google have developed smart-phone driven robots that can learn from each other via the cloud [6]. A research group at Singapore's ASORO laboratory has built a cloud computing infrastructure to generate 3-D models of environments, allowing robots to perform simultaneous localization and mapping (SLAM) much faster than by relying on their onboard computers [7].

Cloud Robotics

In this section, we first describe a system architecture for cloud robotics, and then focus on the two key enabling subsystems: the M2M/M2C communication framework and the elastic computing architecture. Our cloud robotics differentiates from existing solutions in that it leverages two complementary clouds (i.e., an ad-hoc cloud and an infrastructure cloud).

System Architecture

In Fig. 2, we illustrate the system architecture for our proposed cloud robotics. The architecture is organized into two complementary tiers: a machine-to-machine (M2M) level and a machine-to-cloud (M2C) level.

On the M2M level, a group of robots communicate via

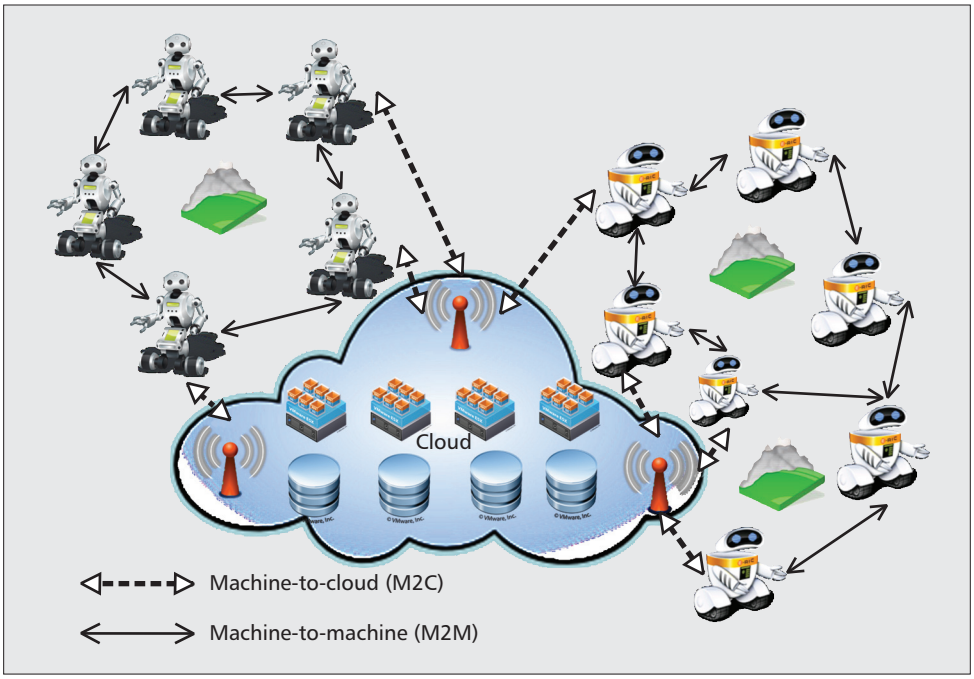

Figure 2. *Cloud Robotics: robots are interconnected via M2M/M2C communications, sharing their resources and accessing to remote cloud resources.*

wireless links to form a collaborative computing fabric (i.e., an ad-hoc cloud). The benefits of forming a collaborative computing fabric are multi-fold. First, the computing capability from individual robots can be pooled together to form a virtual ad-hoc cloud infrastructure. Second, among the collaborative computing units, information can be exchanged for collaborative decision making in various robot-related applications. Finally, it allows robots that are not within communication range of a cloud access point to access information stored in the cloud infrastructure or send computational requests to the cloud.

On the M2C level, the infrastructure cloud provides a pool of shared computation and storage resources that can be allocated elastically for real-time demand. This elastic computing model allows the group of networked robots to offload computation-intensive tasks for remote execution, resulting in "remote-brain" robots. Moreover, the benefits of a large volume of storage provided by the centralized cloud are two-fold. First, it can unify a large volume of information about the environment, which can be organized in a format usable by robots. Second, it can provide an extensive library of skills or behaviors that are related to task requirements and situational complexities, making it feasible to learn from the history of all cloud-enabled robots.

M2M/M2C Communication Architecture

Robots in a network can communicate if they are within communication range of each other, and with the cloud servers if the robots are close to access points of the cloud infrastructure. A wireless M2M communication network can be formed by robots working cooperatively with each other to route and relay information. We call either a robot or an access point a node in the M2M/M2C network.

Several standards like Zigbee, Bluetooth, and WiFi Direct have been developed for short range wireless communications between robots. For long range communications, radio frequency and microwave communication technologies may be used.

A network of robots is often formed dynamically and in an ad-hoc manner. There is no central controller to coordinate the communication flow in the network. Robots may leave and join the network, or may become unavailable because of unpredictable failures or obstructions in the environment. Furthermore, the network is highly dynamic if robots are mobile. All these considerations make the design of effective routing protocols difficult and impractical in some scenarios. Gossip algorithms [8] are randomized methods designed to transmit a message from a source to a destination without any explicit route discovery mechanisms. If two nodes are within communication range, we say that they are neighbors. When a robot wants to send a message to a destination node (either another robot or a cloud access point), it randomly chooses one of its neighbors and transmits the message together with a header that contains the identifier of the destination and itself, and a time value indicating the validity period of the message. In another variant of the protocol, the message is simply broadcast to all neighbors, but depending on the application, this may incur high communication load in the network. We will focus on the protocol that chooses a random neighbor in this article. At every time step, each node randomly chooses a neighbor to retransmit messages that are not intended for itself and are still valid. After a sufficient number of time steps, all the messages will be relayed to their destinations with high probability.

We propose the use of gossip protocols for M2M/M2C communications in cloud robotics. Gossip protocols do not require route discoveries and maintenance, and are thus suited for highly dynamic mobile robotic networks. These protocols are also very simple to implement, and require minimal additional computation and memory resources. However, the trade-off is that gossiping may result in a high message latency if the network conductance [8] is low. In cloud robotics, however, this problem is significantly mitigated as the cloud backend infrastructure serves as a central super node for the M2M/M2C communication network. As we will see, the time required for a message to be disseminated in a network is greatly reduced by the existence of a super node. Alternatively, a hybrid gossip algorithm can be used in which routes to frequently accessed nodes like the group leader in the proxy-based cloud computing model, can be maintained.

Elastic Cloud Computing Architecture

Our proposed cloud robotics is built on the combination of an ad-hoc cloud formed by a group of networked robots and an infrastructure cloud. This unique combination offers us great flexibilities in designing computing models tailored for specific applications. We focus on the following three elastic computing models (Fig. 3):

- *Peer-Based Model:* each robot or virtual machine (VM) in the ubiquitous cloud is considered as a computing unit. These robots and VMs form a fully distributed computing mesh. A task can be divided into smaller modules for execution over a subset of the nodes in the computing mesh.
- *Proxy-Based Model:* in the group of networked robots, one unit functions as a group leader, communicating with a proxy VM in the cloud infrastructure, to bridge the interaction between the robotic network and the cloud. The set of computing units are organized into a two-tier hierarchy.
- *Clone-Based Model:* each robot has a corresponding system-level clone in the cloud. A task can be executed in the robot or in its clone. The set of robotic clones also form a peer-to-peer network with better connectivity than the physical ad-hoc M2M network. Moreover, this model allows for sporadic outage in the physical M2M network.

Each of these elastic computing models exhibits different robustness in network connections, interoperability, and mobility flexibility (Table 1). Robustness refers to the network connectivity between the set of networked robots and the infrastructure cloud . The clone-based model has the maximum number of linkages from robots to the infrastructure cloud, and is thus the most robust; the proxy-based model is the least robust in terms of network connectivity; and the peer-based model falls between these two extreme cases. Interoperability refers to the additional complexity required in operating a cloud robotics infrastructure with an existing robotic network. The proxy-based model is the most interoperable model, because of its hierarchical structure; while the clone-based model is the least interoperable. Mobility flexibility refers to the network's ability to support mobile robots. The peer-based model has the most flexibility for mobility, because VMs can be instantiated anywhere in the cloud infrastructure; while the clone-based model is the least flexible, because complicated VM migration mechanisms are required to support robot mobility.

The choice of a specific elastic computing model depends mainly on three factors, including network conditions, application requirements and resource availability. We aim to develop a unified framework to determine an optimal or near-optimal model for a given set of conditions, as elaborated later.

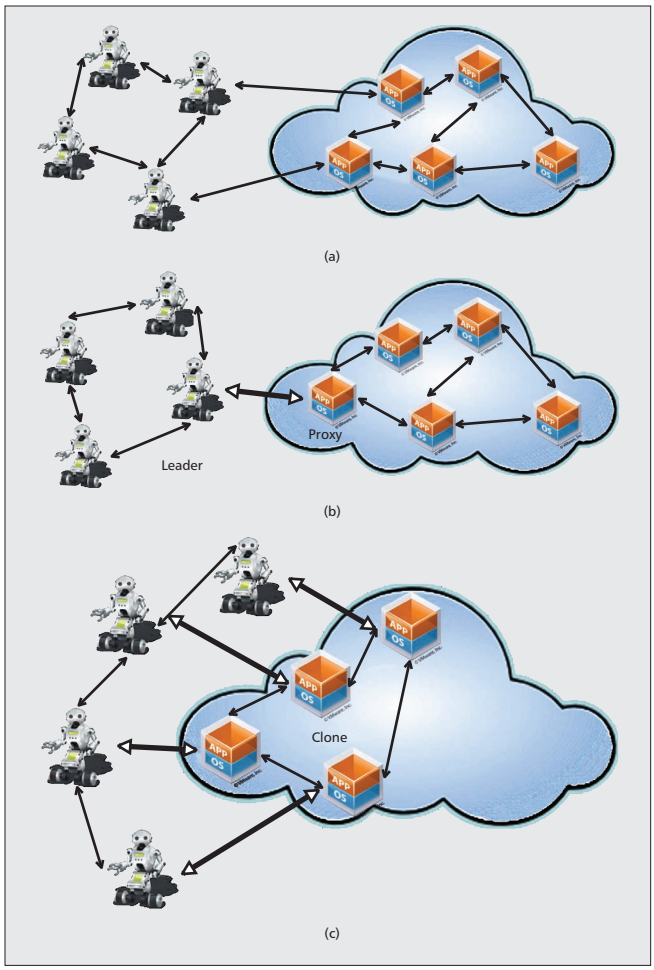

Figure 3. *Elastic computing models for cloud robotics: a) peer-based model; b) proxy-based model; c) clone-based model.*

Technical Challenges

In this section, we discuss some specific challenges and research considerations in the elastic computation model and the M2M/M2C communication network.

Computation Challenges

One of the key benefits of cloud robotics is the capability of offloading computationally intensive tasks to the cloud for execution. However, the decision to offload a specific task requires a unified framework that can handle a list of complex issues. First, the offloading strategy should consider various factors, including the amount of data exchanged, and the delay deadline to complete the task. Second, the decision should also consider whether it is more advantageous to execute the task within the group of networked robots, given the presence of cloud resources. Finally, given a pool of cloud resources spread across different data centers, it is a challenge to allocate virtual machines optimally to execute the offloaded task and to manage live VM migrations.

We advocate an unified optimization framework to determine the optimal task execution strategies. Specifically, our objective is to minimize the amount of energy consumed by the robot, under the constraint that the task should be completed within a specified deadline. The fundamental trade-off lies between the energy consumed for executing the task by the on-board CPU within the robot and the energy consumed transmitting the amount of data to the cloud for remote execution.

In our initial investigation, we have considered two alternative choices of a standalone execution by the robot and a cloud execution. We adopt the following energy consumption model. For the standalone execution, Dynamic Voltage Scaling (DVS) [9] is assumed to minimize the total energy usage for the computing task; for the cloud execution, we assume a polynomial energy consumption model in which the amount of energy consumed to transmit s bits across a wireless channel with fading coefficient g is proportional to s^n/g, where n depends on the coding scheme. It can be shown that, for a given task profile of L bits of data and a delay deadline of T, the minimum energy consumed for the standalone execution is $\varepsilon_r^* \sim L(L/T)^2$, and the minimum transmission energy for the cloud execution is $\varepsilon_c^* \sim L(L/T)^{(n-1)}$.

Using the above results, we can determine the optimal operational region for either task execution model by simply

Model	Robustness	Interoperability	Mobility
Peer-based	Medium	Medium	High
Proxy-based	Low	High	Medium
Clone-based	High	Low	Low

Table 1. *Comparisons of different computing models.*

comparing the energy usage for a specific task. For example, the optimal task execution region is illustrated in Fig. 4 for $n = 2.5$. The boundary between the two optimal operational regions is a line (i.e., $L/T = const$), where $R_e = L/T$ can be interpreted as an effective data consumption rate. It corresponds to a threshold policy. In this case, when the effective data consumption rate is larger than the threshold, cloud execution is more energy efficient; otherwise, standalone execution is more energy efficient. In-depth theoretical and numerical analysis for this simple task offloading scheme can be found in our previous work [10], and is omitted here due to limited space.

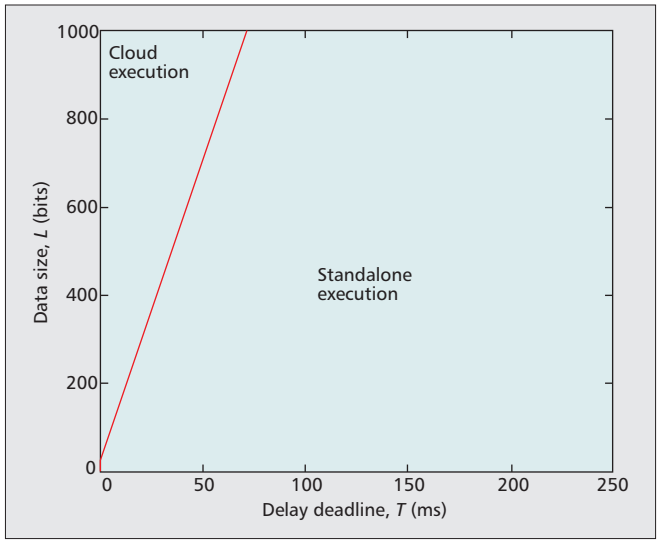

Figure 4. *The optimal operational regions are separated by a line. Its slope corresponds to the effective data consumption rate. In this graph,* n = *2.5.*

Communication Challenges

As discussed in the previous section, the choice of standalone or cloud execution depends on the delay sensitivity of the task. The communication delay introduced in sending the computation request to the cloud has to be factored into the decision. In this section, we provide upper bounds for the time required for a message to be delivered with *high probability*. Packet delivery failures and communication outage are inherent in any wireless communication systems. A communication network based on gossip protocols can thus be regarded as a system with higher failure rates, but significantly lower overheads. Furthermore, the additional increment in the failure rate depends on the network topology, and the use of the infrastructure cloud as a super node in the network effectively controls this rate.

In the gossip protocol, each node chooses a neighbor randomly to transmit a message. Suppose that node i chooses node j with probability P_{ij}, where a zero probability implies that the two nodes are not within communication range, and are therefore not neighbors. It can be shown [8] that the communication delay of disseminating a message from a single node in the M2M network to *all* N nodes in the network is $O(\log N/\Phi)$, where Φ is the conductance of the network, given by

$$\Phi = \min_{S:|S|\le N/2} \frac{\sum_{i\in S, j\in S^c} P_{ij}}{|S|}.$$

In general, the worst case communication delay is $O(N \log N)$. However, in our cloud robotics architecture, M2M links are expected to be short range, so that we can partition the M2M network into connected components of robots with some maximum size M, with each component having at least one link to the cloud super node. For example, in the clone-based model where all robots have communication links to the cloud, the size $M = 1$. Typically M is either constant or grows slowly with the size N of the network, and is much smaller than $N/2$. We see that the conductance of such a network is of order at least $1/M$, so that the delay is bounded by $O(M \log N)$. In addition, task offloading is typically to immediate neighbors or to the cloud. Therefore, in the worst case, the time required for M2C communications is $O(M \log M)$. See Table 2 for the worst-case delays for each computing model. For the peer-based model, the communication delay depends on the particular network topology. We assume that for any subset of nodes in the network, there exists at least a fraction a with communication links to the cloud. Such a network belongs to the class of expander graphs.

The above analysis applies for a static network. We have shown that it generalizes to similar results for mobile networks with switching topologies [11].

Optimization Framework

We have considered the simple scenario of standalone versus cloud execution of a task. In general, the task offloading decision should made among three execution strategies, including:

- The standalone execution by the individual robot,
- The collaborative execution by the group of networked robots, and
- The cloud execution.

In some cases, a hybrid model including partial execution with all these strategies is possible. We aim to develop an optimization framework involving all execution modes with communication and execution costs included, to find the optimal execution strategy. In particular, the optimal strategy should take into consideration the time-varying nature of the wireless M2M/M2C communication network and the latency introduced by the gossiping protocol. In addition, only the minimal required set of information should be communicated, and this depends on the particular application. It is also of interest to investigate what information should be stored locally versus on the cloud.

Security Challenges

Trust and security issues are major considerations in cloud robotics. Specifically, our solution faces two major security challenges due to its cloud implementation.

First, we need the VM environment to be trust-worthy. A malicious VM can subtly sabotage an important task without the robot being aware of the damage. In military applications, the robot has to identify a trust-worthy VM infrastructure to connect and to avoid malicious infrastructures (e.g., battlefield communication vehicles from an enemy). In general, three approaches can be adopted to cope with this problem, including:

- *Trust Establishment:* the user performs some pre-use actions to check a VM's host environment.
- *Trust Measurement:* some root-of-trust components that do

Model	M2M	M2C
Peer-based (expander)	$O(1/\alpha \log N)$	$O(1/\alpha \log N)$
Proxy-based	$O(M \log N)$	$O(M \log M)$
Clone-based	$O(\log N)$	$O(1)$

Table 2. *Comparisons of worst-case communication delays for different elastic computing models.*

not belong to the cloud platform provider (e.g., from hardware vendors or virtualization software providers) monitor the VM, and securely report trust measurements to a user or a third party.

- *Reputation-Based Trust:* the user verifies the VM infrastructure by the service provider's identity and then relies on legal, business or other external considerations to infer trust.

Second, a robot needs trust to launch task delegation on a public cloud, especially when the computation and network traffic incur monetary costs. The computing environments in the cloud should be verifiable by a user or a trusted party, e.g., to ensure there is no hidden or malicious code running besides the delegated tasks. Moreover, confidential data may be stored in the public cloud storage, while logically private to clone devices. Therefore, strong integrity and confidentiality protection are needed to secure application data.

Robotic Applications

Future robotic applications will benefit from cloud robotics, which provides the following advantages over traditional networked robots.

- Ability to offload computation-intensive tasks to the cloud. The robots only have to keep necessary sensors, actuators, and basic processing power to enable real-time actions (e.g., real-time control). The battery life is extended, and the robotic platform becomes lighter and less expensive with easier to maintain hardware. The maintenance of software onboard with the robots also becomes simpler, with less need for regular updates. As the cloud hardware can be upgraded independently from the robotic network, the operational life and usefulness of the robotic network can be easily extended.
- Access to vast amounts of data. The robots can acquire information and knowledge to execute tasks through databases in the cloud. They do not have to deal with the creation and maintenance of such data.
- Access to shared knowledge and new skills. The cloud provides a medium for the robots to share information and learn new skills and knowledge from each other. The cloud can host a database or library of skills or behaviors that map to different task requirements and environmental complexities. The RoboEarth project [12] is trying to turn this into a reality.

Due to these advantages, cloud robotics has a wide range of potential applications in data-intensive or computation-intensive tasks in the areas of intelligent transportation, environment monitoring, health care, smart home, entertainment, education, and defense. In this section, we discuss the opportunities and challenges that cloud robotics brings to traditional robotic applications. Specifically, we focus on three robotic applications: *SLAM, grasping, and navigation*.

SLAM

SLAM [13] refers to a technique for a robot or an autonomous vehicle to build a map of the environment without a priori knowledge, and to simultaneously localize itself in the unknown environment. SLAM, especially vision-based SLAM and cooperative SLAM, are both *data intensive and computation intensive*. The steps such as map fusion and filtering for state estimation can be processed in a parallel fashion. Thus, these tasks can be offloaded to the cloud. For example, a grid based FastSLAM is implemented in a cloud computing framework as reported in [7]. As demonstrated in [7], the cloud can substantially improve the implementation speed of SLAM.

Grasping

Robotic grasping has been an active research topic over a few decades. If the full 3-D model of the object is precisely known, then various methods can be applied to synthesize the grasp. If the object is unknown or not precisely known, the problem is much more challenging, and involves the access and preprocessing of vast amounts of data and can be computationally intensive. Recently, information-based or data-driven grasping methods [14] have been developed to enable robotic grasping for any hand and any object. These methods requires *access to large databases*. By offloading this task to the cloud, grasping can be facilitated without requiring vast amounts of computing power, data, and storage space on the robotic platform. In addition, model knowledge of new objects learned by different robots can be shared in the cloud for future usage by other robots.

Navigation

Robotic navigation refers to a robot's activity to determine its own position with respect to a certain reference and then plan a path to reach a desired location. It can involve a combination of tasks such as localization, path planning, and mapping. Basically, there are two types of approaches: map-less approaches and map-based approaches [15]. Map-less approaches rely on the observations of the perception sensors for navigation. Due to the limited onboard resources, these approaches usually suffer from reliability issues. Map-based robotic navigation is relatively reliable if a precise map is available. It can either use a known map or build a map during the navigation. However, the process of building the map requires large amounts of storage space and is computationally intensive. On the other hand, the process of searching a map requires access to large amounts of data, which is challenging if the navigation area is large. Cloud robotics provides a very promising solution for future cloud-enabled navigation that avoids these two challenges. The cloud can not only provide storage space to store the large amount of map data, but also provide processing power to facilitate the building and searching of the map quickly. Through the cloud, commercially available maps (e.g., Google maps) can also be leveraged to develop reliable, agile, and long-range autonomous navigation solutions.

Conclusions

We have proposed a cloud robotics architecture to address the constraints faced by current networked robots. Cloud robotics allows robots to share computation resources, information and data with each other, and to access new knowledge and skills not learned by themselves. This opens a new paradigm in robotics that we believe leads to exciting future developments. It allows the deployment of inexpensive robots with low computation power and memory requirements by leveraging on the communications network and the elastic computing resources offered by the cloud infrastructure. Applications that can benefit from the cloud robotics approach are myriad and includes SLAM, grasping, navigation, and

many others that we have not discussed, like weather monitoring, intrusion detection, surveillance, and formation control.

References

[1] B. Siciliano and O. Khatib, Eds., *Springer Handbook of Robotics*, Springer, 2008.
[2] IEEE Society of Robotics and Automation's Technical Committee on Networked Robots, available: http://www-users.cs.umn.edu/~isler/tc/
[3] P. Jacquet *et al.*, "Optimized Link State Routing Protocol for Ad Hoc Networks," *Multi Topic Conf. 2001, IEEE INMIC 2001*, Technology for the 21st Century, *Proc. IEEE Int'l.*, 2001, pp. 62–68.
[4] C. Perkins *et al.*, "Performance Comparison of Two On-Demand Routing Protocols for Ad Hoc Networks," *IEEE Personal Commun.*, vol. 8, no. 1, Feb. 2001, pp. 16–28.
[5] P. Mell and T. Grance, "The Nist Definition of Cloud Computing," NIST Special Publication 800-145, Sept. 2011, available: http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf.
[6] Google cloud robotics, available: http://googlemonthly.com/ google-directions/google-io-2011-cloud-robotics.html.
[7] R. Arumugam *et al.*, "DAvinCi: A Cloud Computing Framework for Service Robots," Int'l. Conf. Robotics and Automation, 2010, pp. 3084– 3089.
[8] D. Shah, "Gossip Algorithms," *Foundations and Trends in Networking*, 2008, vol. 3, no. 1.
[9] J. M. Rabaey, Ed., Digital Integrated Circuits, Prentice Hall, 1996.
[10] Y. Wen, W. Zhang, and H. Luo, "Energy-Optimal Mobile Application Execution: Taming Resource-Poor Mobile Devices With Cloud Clones," *Proc. 31st IEEE Int'l. Conf. Comp. Commun.*, Mar. 2012.
[11] D. W. Soh, T. Q. S. Quek, and W. P. Tay, "Randomized Broadcast in Dynamic Network Environments," *Proc. IEEE Int'l. Wksp. Signal Proc. Advances for Wireless Commun.*, June 2011.
[12] M. Waibel *et al.*, "Roboearth — A World Wide Web for Robots," *IEEE Robotics & Automation Mag.*, vol. 18, no. 2, June 2011, pp. 69–82.
[13] H. Durrant-Whyte and T. Bailey, "Simultaneous Localization and Mapping: Part I," *IEEE Robotics & Automation Mag.*, vol. 13, 2006, pp. 99–110.
[14] C. Goldfeder and P. K. Allen, "Data-Driven Grasping," *Autonomous Robots*, vol. 31, no. 1, pp. 1–20, Apr. 2011.
[15] F. Bonin-Font, A. Ortiz, and G. Oliver, "Visual Navigation for Mobile Robots: A Survey," *J. Intelligent and Robotic Systems*, vol. 53, 2008, pp. 263–96.

Biographies

GUOQIANG HU [M] (gqhu@@ntu.edu.sg) is an Assistant Professor in the School of Electrical and Electronics Engineering at Nanyang Technological University in Singapore. Prior to his current position, he was a postdoc research associate at University of Florida in 2008 and an assistant professor at Kansas State University from 2008 to 2011. He received his B.Eng, M.Phil, and Ph.D. degrees from University of Science and Technology of China, the Chinese University of Hong Kong, and University of Florida in 2002, 2004, and 2007, respectively. His research interest is in the analysis, control, and design of distributed intelligent systems.

WEE PENG TAY [M] (wptay@ntu.edu.sg) is an Assistant Professor in the School of Electrical and Electronics Engineering at Nanyang Technological University in Singapore. He received the BS degree in Electrical Engineering and Mathematics, and the M.S. degree in Electrical Engineering from Stanford University in 2002. He received the Ph.D. degree in Electrical engineering and Computer science from the Massachusetts Institute of Technology in 2008. His main research interests are in distributed signal processing and algorithms, data fusion and decision making in ad hoc networks, machine learning and applied probability.

YONGGANG WEN [M] (ygwen@ntu.edu.sg) is an Assistant Professor in the School of Computer Engineering at Nanyang Technological University (NTU) in Singapore. Prior to his present position, he has held R&D positions in networking companies in the USA, including Cisco and Lucent. He received his Ph.D. degree in electrical engineering and computer science from Massachusetts Institute of Technology (MIT) in 2008, his M.Phil. degree in information engineering from Chinese University of Hong Kong (CUHK) and B.Eng. degree in electronic engineering and information science from University of Science and Technology of China (USTC) in 2001 and 1999, respectively. His research interests are in cloud computing, content networking and green networks.